APPLICATION FOR UNITED STATES LETTERS PATENT

FOR

HIERARCHICAL IMAGE FEATURE-BASED VISUALIZATION

BY

ULLAS GARGI
and
DANIEL TRETTER

BURNS, DOANE, SWECKER & MATHIS, L.L.P.
POST OFFICE BOX 1404
ALEXANDRIA, VIRGINIA 22313-1404
(703) 836-6620
Attorney's Docket No. 10006286-1

HIERARCHICAL IMAGE FEATURE-BASED VISUALIZATION

BACKGROUND

Field of the Invention

[0001] The present invention relates to multimedia database and classification systems, and in particular to automatic classification and retrieval of multimedia files based on the features of the multimedia files.

Background Information

[0002] Automatic image classification has many important applications. Large image databases or collections require good indexing mechanisms so that images can be categorized effectively, browsed efficiently, and retrieved quickly. Conventional systems store and retrieve specific information from a database using, for example, descriptive information regarding the image file, such as file creation date, file name, file extension and the like. This form of image classification is not significantly different from the classification of any other digital information.

[0003] By relying on the file information, only cursory information can be obtained about the file and nothing at all specifically related to the image. For example, an image file could have a name that has no relation to the features or content of the image, such as a black and white image could have the file name "color_image". Other systems provide classification based on the content of the images, such as flowers, dogs, and the like. In practice, this is usually done by keyword annotation, which is a laborious task.

[0004] The amount of multimedia information available today due to the evolution of the internet, low-cost devices (e.g., digital video cameras, digital cameras, video capture cards, scanners and the like) to generate multimedia content, and low-cost storage (e.g., hard disks, CDs, and the like) increases the need to classify and retrieve relevant multimedia data efficiently. Unlike text-based retrieval, where keywords are successfully used to index into documents, multimedia data retrieval has no easily accessed indexing feature.

[0005] One approach to navigating through a collection of images for the purpose of image retrieval is disclosed by Yossi, R., "Perceptual Metrics for Image Database Navigation," PHD Dissertation, Stanford University May 1999, which is incorporated herein by reference in its entirety. The appearance of an image is summarized by distributions of color or texture features, and a metric is defined between any two such distributions. This metric, called the "Earth Mover's Distance" (EMD), represents the least amount of work that is needed to rearrange the images from one distribution to the other. The EMD measures perceptual dissimilarity which is desirable for image retrieval.

Multi-Dimensional Scaling (MDS) is employed to embed a group of images as points in a 2- or 3-dimensional (2D or 3D) Euclidean space so that their distances reflect the image dissimilarities. This structure allows the user to better understand the result of a database query and to refine the query. The user can iteratively repeat the process to zoom into the portion of the image space of interest.

[0006] Feature extraction is a key component to generating systems that can organize multimedia files based on their content. References that address image feature extraction include the following articles. Niblack, et al., "The QBIC Project: Querying Images by Content Using Color, Texture, and Shape," Proc. of SPIE, Storage and Retrieval for Image and Video Databases, Vol. 1908, February 1993, San Jose, pp. 173-187, which describes using color histograms for image distance measurement and is hereby incorporated by reference. M. J. Swain and D. H. Ballard, "Color Indexing," International Journal of Computer Vision, Vol. 7, No. 1, pp.11-32, 1991, which describes histogram intersection techniques and is hereby incorporated by reference. G. Pass and R. Zabih, "Histogram Refinement for Content-based Image Retrieval," IEEE Workshop on Applications of Computer Vision, pp. 96-102, 1996, which describes a color coherence vector and is hereby incorporated by reference. J. Huang, et al., "Image Indexing Using Color Correlogram," IEEE Int. Conf. on Computer Vision and Pattern Recognition, pp. 762-768, Puerto Rico, June 1997, which describes the use of color correlograms as features in indexing images and is hereby incorporated by reference. H. Tamura, S. Mori, and T. Yamawaki, "Texture Features Corresponding to Visual Perception," IEEE Trans. On Systems, Man, and Cybernetics, vol. 8, no. 6, June 1978, which describes the use of texture as features in images processing and is hereby incorporated by reference. M. K. Hu, "Visual Pattern Recognition by Moment Invariants," IEEE computer Society, Los Angeles, CA, 1977, which describes the use of moment invariants as features in

images processing and is hereby incorporated by reference. J. Mao and A. K. Jain, "Texture Classification and Segmentation Using Multiresolution Simultaneous Autoregressive Models," Pattern Recognition, Vol. 25, No. 2, pp. 173-188, 1992, which describes Multiresolution Simultaneous Autoregressive Model (MRSAR) texture features and is hereby incorporated by reference.

[0007] Since visualizing and retrieving large databases of multimedia files are complex tasks, it is desired to have a method for visualizing and retrieving data files that provides a distance calculation that starts at a coarse level of feature differences and progressively increases to a fine level of feature differences as the number of data files displayed is decreased. Additionally, it is desired to have an interactive real time system that allows a user to interactively select portions of the displayed data files to search and retrieve data files from large databases.

SUMMARY OF THE INVENTION

[0008] The present invention is directed to methods and systems for visualizing and retrieving data files. An exemplary method comprises: displaying a plurality of images representing data files on a display device using a first distance metric between each data file; redisplaying a portion of the images on the display device using a refined distance metric; and retrieving the desired data file.

[0009] An exemplary method of interactively retrieving data files in real time comprises: displaying plural images, each image corresponding to a data file, on a display device using a first distance metric between each data file; interactively selecting, by a user, a portion of the images; redisplaying the portion of the

images in real time on the display device using a refined distance metric; and retrieving a desired data file.

BRIEF DESCRIPTION OF THE DRAWINGS

[0010] The above features and advantages of the invention, and additional features and advantages of the invention, will be better appreciated from the following detailed description of the invention made with reference to the drawings, wherein:

FIG. 1 shows a flow chart of an exemplary method of the present invention;

FIGS. 2A-2E show screen captures of progressive levels of displays generated by the present invention;

FIG. 3 shows a flow chart of another exemplary method of the present invention;

FIG. 4 shows a flow chart of performing a coarse to fine distance calculation of the present invention; and

FIGS. 5A and 5B show sample images that correspond to the feature vectors of Appendix A.

DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENTS

[0011] FIG. 1 is a flow chart of an exemplary method of visualizing and retrieving data files. The method begins in step 110 by displaying a plurality of images representing data files on a display device using a first (i.e., coarse) distance metric between each data file. According to alternate exemplary

embodiments, the data file can be a segment of a large data file. In step 120, a portion of the images can be redisplayed on the display device using a refined distance metric. Optionally, in step 130, the redisplaying step can be repeated until a desired data file is identifiable. The distance metric can be refined further at each redisplay until the desired data file is found or a maximum refined distance metric (e.g., all image feature data are used to calculate the distance metric) is reached. The desired data file can be retrieved in step 140. Alternatively, the desired data file can be marked, selected, and the like, as will be appreciated by those skilled in the art. For instance, the desired data file can be marked for later processing by another program. Furthermore, those skilled in the art will appreciate that many additions and variations can be added to the above process steps while still keeping within the scope of the invention. Some of these alternate embodiments will be described in the following text.

[0012] FIGS. 2A-2E show a graphical representation of displays generated by the present invention, where each portion of the images redisplayed, as shown in the subsequent figure, can be graphically selected by a user. FIG. 2A is a screen capture of an exemplary display showing a first level of the images representing data files on a two-dimensional display. There are more than one thousand images displayed in FIG. 2A. As can be seen in FIG. 2A, the images are not very distinguishable and there are many images that are obscured due to overlapping of the images. Therefore, generating a precise distance between images is not advantageous, because the finer details that separated images are indistinguishable

at this resolution. Therefore, a first (i.e., coarse) distance metric can be calculated. However, the first distance metric can still allow the user to receive useful information about the organization of the images on the display. For instance, there are clear trends from dark to light colors going from left to right. Another visible trend is from top to bottom, where the images tend to go from softer images to images with distinct edge lines.

[0013] Based on the perceived properties of a desired image, a user can select the area 202 of the screen where the desired image most likely resides. Those skilled in the art will appreciate that this selection process can be accomplished by any technique known in the art, such as using a mouse, touch screen, stylus, screen coordinates, keystroke combination, and the like. Additionally, the portion of the images redisplayed can represent images selected from noncontiguous regions of the display. For example, the images can be selected from area 202 and another region 204 located on the lower portion of the display. Alternatively, individual images can be selected. In each case, however, not only are the visible images selected, but all images that are mapped within that space (e.g., the overlapped images).

[0014] The overlapping of the center of the display can be several hundred images deep. Therefore, in an alternative embodiment, a display depth indication can be provided that represents an amount of overlapping of images on the display. In combination with the depth display, the user can scroll in and out of the display to view images that were previously not viewable due to the

overlapping of the images. For example, a scroll wheel of a mouse can be used to navigate the depth of the images. However, those skilled in the art will appreciate that any known technique for scrolling on a display can be utilized. Additionally, the images could be mapped into a three or higher dimensional space and navigated using virtual reality software and hardware as is known in the art.

[0015] Still further, a fixed scale can be established that spans a maximum distance between the plurality of data files. A relative position indicator can also be provided on the fixed scale for each redisplay of the portion selected, thereby providing the user with a reference frame at each level of redisplay. Thus, as the redisplay level increases and the relative separation of the images increases (e.g., see FIGS. 2B to 2D), a user can have a reference that relates back to the first level of the display. The fixed scale can be a linear scale, a logarithmic scale, or a hyperbolic scale. For example, if the distance metric is large between the images, then nonlinear scaling can be used.

[0016] Dimensions displayed along the X and Y axes at successive display steps can be different, because the dimensionality reduction process can be run on two different set of data-vectors (e.g., images). However, one or both of the dimensions displayed along the X and Y axes can be fixed to increase coherence with the previous display, so that the movement of images is reduced. For example, fixing both dimensions to be the same can be considered the equivalent to performing a simple geometrical zoom operation.

[0017] FIG. 2B shows a portion of the images redisplayed as selected by area 202. Seventy-eight images are included in this redisplay using a refined distance metric. The distance metric can be recalculated using more of the image information (i.e., feature data) than was used in the first distance calculation. Thus, the reclustering and redisplaying of the selected images are more than merely a "zoom" function. Once again, a user can select a portion 210 of the images at this level for further redisplay. This process can be repeated as shown in FIGS. 2C and 2D. Specifically, FIG. 2C, which is based on the portion 210, contains fifty-one images. The distance metrics between the images at this level can be again recalculated using even more of the image feature data than previously used in relation to FIG. 2B. A portion 220 of the images displayed at this level is selected. FIG. 2D, which is based on portion 220, contains twenty-three images. The distance metrics between the images at this level can be again recalculated using even more of the image feature data than previously used in relation to FIG. 2C. A portion 230 of the display at this level can be further selected. However, at this level the desired data file (i.e., image 240 at the lower portion) is identifiable. Thus, it can be selected and retrieved as shown in FIG. 2E. In this example, the desired data file is an image file. However, the data files can also be video files. In either case, the images that are displayed can be representations of the underlining data file (e.g., a thumbnail image, or a 3D image of an entire video segment) that require less data processing to generate the display.

[0018] The previous description provides a basis for understanding the interactivity of the present invention. FIG. 3 shows a flow chart for a method of interactively retrieving data files in real time. In step 310, the process starts by displaying plural images, each image corresponding to a data file, on a display device using a first (i.e., coarse) distance metric between each data file. In step 320, a user interactively selects a portion of the images displayed. The selected images can be redisplayed in real time on the display device using a refined distance metric, in step 330. Optionally, the selecting and redisplaying steps can be repeated until a desired data file is identifiable, in step 340. Those skilled in the art will appreciate that the distance metric can be further refined at each level of redisplay until the maximum refinement of the distance metric is reached or the desired data file is identified. However, the interaction process of the present invention is not limited to the level of maximum refinement of the distance metric. For example, if the refinement only supports a certain number of levels, a user may still browse through additional levels beyond that which is supported. However, these additional levels will use the same distance metric as that supported at the level of maximum refinement of the distance metric.

[0019] In step 350, the desired data file can be retrieved. Alternatively, the desired data file could be marked, selected, and the like. Those skilled in the art will appreciate that processing large multimedia databases in real time consumes significant processing power. The prior art systems, such as the previously described Yossi system, rely on complex algorithms, such as Multi-Dimensional

Scaling, for mapping the image feature vectors to a 2D space. These systems and algorithms are too slow to work in a real time interactive system, such as the present invention.

[0020] FIG. 4 shows a flow chart for an exemplary method of refining a coarse distance metric, which can be used in connection with the methods described in relation to FIGS. 1 and 3. The process begins in step 410 by computing a feature vector for each data file. This feature vector can be calculated in advance of starting the distance metric refining process. For example, the feature vectors for each image in a large image database can be calculated once and stored in a related database. Therefore, in step 410, the "computing" function would be reduced to the computational effort necessary to retrieve the appropriate image feature vectors. Optionally, in step 415, a distance matrix can be calculated that stores the distance between each image file in a k-by-k matrix (see, Table 1 below), where k is the number of data files. In step 420, the first (i.e., coarse) distance metric can be calculated between each data file using a first subset of data contained in the feature vector. If the distance matrix is calculated, then N-dimensional space mapping can be calculated directly from the distances stored in the distance matrix, as described below.

[0021] In step 430, a second subset of data from the feature vector can be established which is greater than the first subset. Once again, a distance matrix can be optionally calculated that stores the distance between each image file to be redisplayed in a matrix, in step 435. In step 440, a second distance metric

between each data file can be calculated using the second subset of data. Optionally, steps 430, 435, and 440 can be repeated for each additional redisplay (i.e., the distance metric will be refined for a third, fourth, etc. subsets, wherein each subsequent subset of data is greater than the previous subset). At each redisplay level, a portion of the images can be redisplayed using a refined distance metric that is calculated from a subset of data contained in the feature vector, which is greater than the previous subset. The coarse to fine distance calculation process can continue until the desired data file is identified (i.e., no more redisplay of images) or the maximum data subset is reached (i.e., the finest resolution for the distance calculation), in step 450. However, the interaction process of the present invention is not limited to the level of maximum refinement of the distance metric. For example, if the refinement only supports a certain number of levels, a user may still browse through additional levels beyond that which is supported. However, these additional levels will use the same distance metric as that supported at the level of maximum refinement of the distance metric.

[0022] Those skilled in the art will appreciate that the length of the feature vector is a function of the data contained therein. Therefore, the length of the feature vector can be controlled by the type and quantity of feature data used to distinguish the data files. For example, simple images can be distinguished by a color histogram divided into eight separate bins. The ability to distinguish images would be limited to a comparison of the color histograms of each image. Thus,

the distance between each image can be calculated using the eight values of the color histogram for each image. Similarly, mapping distances between each data file to N-dimensional space could be any N value. For example, N can be two or three for 2D and 3D display systems, respectively.

[0023] As previously noted, the length of the feature vector is a function of the type and quantity of the feature data contained therein. The feature vector can include various feature data related to the data file, for example, color and texture. The selection of the feature data contained in the feature vector can be based on the data files and the system designer's preferences. For example, if the image database contains similar color composition (e.g., images of the lunar surface), then the system can use feature data other than color, such as texture, to distinguish between the images (i.e., establish a distance metric between each of the images). However, a general purpose system can use multiple features, such as a feature vector including at least one of a color histogram, color moment, color coherence histogram, Multiresolution Simultaneous Autoregressive (MRSAR) Model, coarseness, and directionality.

[0024] Those skilled in the art will appreciate that the perception of each of these features is not uniform. For example, people are typically more sensitive to color and contrast variation than to texture. Therefore, the feature data can be weighted to reflect the relative perceptibly indicated by that feature. For example, color distance values will be weighted more heavily than texture distance values.

[0025] Referring to FIGS. 5A and 5B, images of rock formations that are similar are shown. Associated with these images is Appendix A that contains descriptions and values of an exemplary feature vector for each image. Each feature vector contains two-hundred-thirty-one values and six features describing each image. For example, the first sixty-four data values correspond to Hue Saturation Intensity (HSV) color space represented in a sixty-four bin color histogram. The next six values relate to first and second order color moments in Lab color space (i.e., two orders by three color channels). The next one hundred twenty eight values are a color coherence vector containing a sixty-four bin histogram with sixty-four coherence bins in LUV space. The next fifteen values are MRSAR texture features. The MRSAR texture features of the image can be computed (with resolution levels 2, 3, 4) on overlapping 21x21 subwindows. Tamura coarseness features are the next ten values in the form of a histogram. The ten bin histogram of coarseness values is computed at 1x1, 3x3, 5x5, 9x9, 13x13, 17x17, 21x21, 25x25, 29x29, and 33x33 window sizes. Finally, a Tamura directionality feature is the last eight values. The Tamura directionality features are computed in sixteen directions and represented as an eight bin histogram. However, these features and associated data are not intended to limit the invention, but are only provided as an example of the features and related feature data contained in an exemplary feature vector.

[0026] The feature vector can contain a significant amount of detailed data about each data file. However, the goal of the feature vector is to represent the data file

in a manner to distinguish each data file from the other data files. Additionally, the feature vector can be used to calculate distances between each data file that corresponds to a person's perception of that data file (e.g., a dark image is closer to other dark images and farther from a light image). Thus, for k images, a k-by-k distance matrix can be formed. Each cell in the matrix represents the distance between corresponding images. A single matrix can contain distance values between all images in a database. An example of an image distance matrix is given below in Table 1. Only the upper or lower half values of the matrix need to be calculated as the values will repeat in the other half (i.e., d_{1-2}, the distance from image 1 to image 2 is the same in both halves).

Image Number	1	2	3	...	k
1	0	d_{1-2}	d_{1-3}	...	d_{1-k}
2	-	0	d_{2-3}	...	d_{2-k}
3	-	-	0		d_{3-k}
...	...	...	...	0	...
k	-	-	-	...	0

Table 1

[0027] Those skilled in the art will appreciate that the first (i.e., coarse) distance measurement can be calculated using less than the total amount of data available in each image feature vector (i.e., a first subset of data contained in the feature vector). At the first level, all of the images in the image database can be displayed, as shown in FIG. 2A. At this level, performing a detailed distance

calculation is not beneficial as both the display resolution and the user's perceptual ability do not permit a fine resolution of the distance measurement. For example, when displaying the over on thousand images in FIG. 2A, just gross trends (e.g., dark to light, dominant colors, etc.) of the images are perceptible in the display. Therefore, the first coarse subset can include only the 20% most dominant colors when calculating the distance metric. As the levels of redisplay increase, the percentage of data used to calculate the distance metric increases (i.e., refining the distance metric). For instance, in a five level system the percentage of color histogram data used can increase as shown in Table 2.

Level	Value
1	20%
2	40%
3	60%
4	80%
5	90%

Table 2

In the above example, the amount of color histogram data used to calculate the distance metric between each image stops at 90% because the lowest 10% of the color histogram does not significantly add to the quality of the distance calculation. Also, the percentages of data used can be adjusted for different features in the feature matrix just as the different features can be assigned different weights. Alternatively, some features can be excluded at the coarse level distance calculation and included in the finer levels (e.g., MRSAR texture features are not

used at first display level, but are used in subsequent redisplay levels). Those skilled in the art will appreciate that the invention is not limited to these techniques. Many other techniques can be used to progressively increase the subset of data used from the feature vector for calculating distances as the redisplay level increases.

[0028] In relation to the feature vectors in Appendix A, an example of weight factors for the various features is provided in Table 3.

Feature	Weight Factor
HSV Histogram	10.05
Color Moment	0.15
Color Coherence	10.50
MRSAR Texture	0.00001
Coarseness	10.20
Directionality	0.10

Table 3

[0029] Table 4, below, shows the results of the distance calculation between image 1 as shown in FIG. 5A and image 2 as shown in FIG. 5B, using the feature values provided in Appendix A, the percentages of Table 2 and the weight factors of Table 3.

Color Histogram Distance at level 1	991522764.0
Color Histogram Distance at level 2	1150625795.0
Color Histogram Distance at level 3	1058898559.0
Color Histogram Distance at level 4	1068628863.0
Color Histogram Distance at level 5	1066543477.0
Color Histogram Distance at level 6	1066543477.0

Color Histogram Distance at level 2	1150625795.0
Color Histogram Distance at level 3	1058898559.0
Color Moments Feature Distance	1018894535.0
Color Coherence Feature Distance	1024732248.0
MRSAR Texture Feature Distance	917510700.0
Coarseness Feature Distance	1034627811.0
Directionality Feature Distance	1012530938.0
Total Feature Distance (level 1)	1042418352.0
Total Feature Distance (level 2)	1150627072.0
Total Feature Distance (level 3)	1061514170.0
Total Feature Distance (level 4)	1069936669.0
Total Feature Distance (level 5)	1067851283.0
Total Feature Distance (level 6)	1067851283.0

Table 4

[0030] Referring to Table 4 and Appendix A, the color histogram distance can be calculated using only a specified percentile of pixels (e.g., Table 2) for comparison at each level. Assume 20% is specified for a given level, then only histogram bins filled by the top 20% most dominant pixels are used. For example, in calculating the color histogram distance, the multilevel distance measure can be used for the first few levels (i.e., given a specified percentile of pixels to use for comparison, such as 20%, only the histogram bins filled by the top 20% most dominant colors are used). This can be performed by sorting the bins of both histograms in descending order. Then, for the histogram in each image feature vector (see, e.g., Appendix A), start from the largest-valued bin and keep marking bins until the specified percentage of pixels is exceeded. Unmarked bins are not used further. Next, reset all marked bins in the histograms to an 'unmatched' status. For each bin in the first histogram, compare it with all

unmatched bins in the second. Match it with the best matching bin (e.g., closest in color features) in the second histogram. Mark both as being 'matched'. This procedure can be repeated until all bins in the first histogram are matched or there are no unmatched bins left in the second histogram.

[0031] Those skilled in the art will appreciate that there are other ways to implement a multilevel distance measure for histograms. For example, a cumulative histogram can be used. Still further, the number of common bins can simply be counted (e.g., common dominant colors occurring in the top 20% bins of the two histograms).

[0032] At deeper levels, a more refined color histogram distance calculation can be performed, as described in the following pseudo code.

```
for (i=0;i<n;i++) {
   result += ((double)(a[i]-b[i])*((double)(a[i]-b[i])));
}
result = sqrt(result);
```

In the above pseudo code, n is the number of bins in the color histograms, a[i] is the value of the ith bin in a first feature vector, and b[i] is the value of the ith bin in a second feature vector. Thus, a Euclidian distance value can be calculated between the color histograms of the first and second feature vectors.

[0033] The color moment distances can be determined by a weighted distance calculation that is used for the 6-element moment vector. The color moments can be calculated as illustrated in the following pseudo code.

```
double weight[6] = {10.2994, 5.0814, 6.8968, 4.2788, 11.2845,
4.9133};
result = 0.0;
for (i=0;i<n;i++) {
        result += fabs((double)(a[i]-b[i])) / weight[i];
}
result /= (double)n;
```

In the above pseudo code, n is the number of elements in the color moment vectors, a[i] is the value of the ith color moment element in a first feature vector, b[i] is the value of the ith color moment element in a second feature vector, and weight[i] is a weighting factor for each element. Thus, a weighted absolute distance value can be calculated between the color moments of the first and second feature vectors.

[0034] The color coherence distance can be calculated, as illustrated in the following pseudo code.

```
for (i=0;i<n;i++) {
        result += ABS((double)(a[i]-b[i]));
}
result /= (double)n;
```

In the above pseudo code, n is the number of elements in the color coherence vectors, a[i] is the value of the ith color coherence element in a first feature vector, and b[i] is the value of the ith color coherence element in a second feature vector. Thus, an absolute distance value can be calculated between the color coherence elements of the first and second feature vectors.

[0035] The MRSAR Texture distance can be calculated as a weighted distance. An example of the pseudo code to accomplish this is given below.

```
double weight[15] = {4.9027, 4.9030, 0.0458, 0.0458, 6.4940,
                     3.5638, 5.6881, 4.4322, 0.0380, 7.1009,
                     0.0713, 0.0772, 0.0342, 0.0344, 7.5016};
result = 0.0;
for (i=0;i<n;i++) {
        result += fabs((double)(a[i]-b[i])) / weight[i];
}
result /= (double)n;
```

In the above pseudo code, n is the number of MRSAR Texture elements in the feature vectors, a[i] is the value of the ith MRSAR Texture element in a first feature vector, b[i] is the value of the ith MRSAR Texture element in a second feature vector, and weight[i] is a weighting factor for each element. Thus, a weighted absolute distance value can be calculated between the color moments of the first and second feature vectors.

[0036] The coarseness distance can be calculated as illustrated in the following pseudo code.

```
for (i=0;i<n;i++) {
        result += ABS((double)(a[i]-b[i]));
}
result /= (double)n;
```

In the above pseudo code, n is the number of elements in the coarseness vectors, a[i] is the value of the ith coarseness element in a first feature vector, and b[i] is the value of the ith coarseness element in a second feature vector. Thus, an

absolute distance value can be calculated between the coarseness elements of the first and second feature vectors.

[0037] Similarly, the directionality distance can be calculated as illustrated in the following pseudo code.

```
for (i=0;i<n;i++) {
        result += ABS((double)(a[i]-b[i]));
}
result /= (double)n;
```

In the above pseudo code, n is the number of elements in the directionality vectors, a[i] is the value of the ith directionality element in a first feature vector, and b[i] is the value of the ith directionality element in a second feature vector. Thus, an absolute distance value can be calculated between the directionality elements of the first and second feature vectors.

[0038] Once again, these values are for example only and not limitation. Those skilled in the art will appreciate that many factors can influence the specific distance calculation technique, such as the weighting of variables. For example, the user can increase the weighting to emphasize desired image features (e.g., coarseness). In an alternate exemplary embodiment, the distance matrix can be maintained as, for example, six different matrices, one for each feature type (color, texture, coarseness, and so forth), along with the corresponding weights. In such an alternate embodiment, the user can change the relative weighting dynamically.

[0039] The Total Feature Distance values provided in Table 4, can be used to populate the distance matrix of Table 1 for each level (i.e., the distance d_{1-2} is 1042418352.0 at the first level, 1150627072.0 at the second level, etc.). The above calculations can be repeated to calculate the distance between every image to complete the image distance matrix.

[0040] The process of calculating coarse to fine distance metrics described above reduces the amount of processing power necessary at the lower levels, because the distance calculations are performed on subsets of the feature vector that contain less data at the lower (i.e., coarse) levels. Although the subset of data from the feature vector used to calculate the distance metrics increases at each subsequent level, the number of images decreases at each subsequent level. Therefore, the process is still more efficient than if the entire feature vector is used at each level.

[0041] To further improve the processing time, particularly for real time interactive systems, mapping the distances into N-dimensional space (e.g., 2D space) can be calculated using FastMap. FastMap is an algorithm for indexing and visualization of multimedia data files. Particularly, FastMap is an efficient algorithm for calculating distances from high dimensional space (e.g., a distance matrix of N-dimensional vectors) to low dimensional space (e.g., a distance matrix for 2D space) while preserving the relative distance relationships. The distance matrix, as originally calculated, contains distances computed between each pair of high-dimensionality vectors. The size of the distance matrix does not correspond

to the dimensionality of the underlying data. For example, each vector can have a length of ten (i.e., ten dimensions). With 200 such vectors, the matrix would be 200x200. FastMap reduces the distance matrix to a distance matrix of the same size (in this example, 200x200) containing distances computed in a lower-dimensional space (for example, 2D or 3D space), with the discarded dimensions being thrown out as residues.

[0042] FastMap is described in an article by C. Faloutsos and L. King-Ip, "FastMap: A Fast Algorithm for Indexing, Data-Mining and Visualization of Traditional and Multimedia Datasets", Proceedings of the 1995 International Conference on Management of Data, 1995, pages 163-174, which is hereby incorporated by reference in its entirety. The FastMap algorithm can be applied to the present invention at each level of redisplay. For example, the first distance metrics can be mapped into an N-dimensional (e.g., 2D) space for the first display. Then, the refined distance metrics can be mapped into an N-dimensional space for redisplaying at each subsequent level.

[0043] The foregoing has described principles, preferred embodiments and modes of operation of the invention. However, the invention is not limited to the particular embodiments discussed above. For example, the examples above described the present invention in the context of image files. However, those skilled in the art will appreciate that the invention can be used on video files. For instance, a frame by frame analysis of the video file can be performed to generate the feature vector. Therefore, the above-described embodiments should be

regarded as illustrative rather than restrictive, and it should be appreciated that variations may be made in those embodiments by those skilled in the art, without departing from the scope of the invention as defined by the following claims.